C Programming Project

Project Report

1.Project Title – Command Line Quiz Game

Submitted By:- Sharon Singh

Sap Id:- 590025912

Course Title: Programming in C

Course Code:- CSEG1041

Semester:-1

Batch:-48

Submitted To:-Mohsin Furkh Dar

2. Abstract

The Command-Line Quiz Game is a simple but interactive console-based application developed using the C programming language. The purpose of this project is to allow users to attempt a series of multiple-choice questions and receive instant feedback along with their final score.

The game uses functions, conditional statements, loops, character handling, and input validation to ensure correct user input. The program uses toupper(), buffer clearing, and modular functions to prevent invalid inputs such as multiple characters or wrong data types.
This project demonstrates how user interaction, decision-making, and logical flow can be implemented effectively in C programming. It also showcases clean input handling and function-based program structure.

3.Problem Definition

Many beginner-level applications do not validate user input properly, leading to crashes, infinite loops, or incorrect outputs. Simple quiz programs often accept invalid input such as numbers, strings, or multiple characters, making them unreliable.

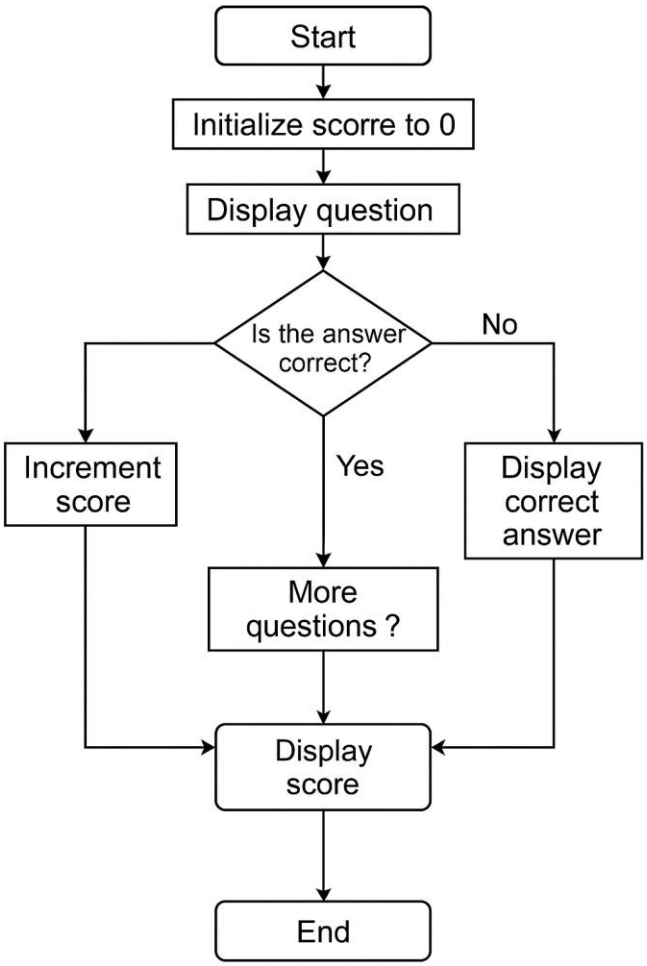
The problem is to design a robust quiz system that:

- Accepts only valid choices: A, B, C, or D
- Prevents invalid input and clears the input buffer
- Displays questions clearly
- Checks answers and keeps score
- Provides a clean, interactive user experience

This project aims to build a safe and user-friendly quiz game in C using functions, validation techniques, and basic logical structures.

4. System Design and Algorithms

. Flow chart



. Algorithms

1. Algorithm: Get Valid Answer

1. Loop forever
2. Ask user for input
3. Read one character
4. Convert to uppercase
5. If character is A/B/C/D → return it
6. Else print error message and repeat

2. Algorithm: Display Question & Check Answer

1. Print question and choices
2. Call getvalidanswer()
3. Compare returned answer with correct option
4. If correct → increment score
5. Else → show correct answer
6. Move to next question

3. Algorithm: Clear Buffer

1. Read characters until newline
2. Discard all remaining input
3. Prevents invalid input issues

4. Algorithm: Final Score Display

1. Print total correct answers
2. If 5 → Excellent
3. If 3–4 → Good Job
4. Else → Better luck next time

5. Implementation Details

Technologies / Concepts Used

- C Programming Language
- Functions
- Conditional statements
- Loops
- Character validation
- toupper() from <ctype.h>
- Input sanitization using buffer clearing
- Modular programming

Code Snippets

• Buffer Clearing Function

```c
void clearBuffer()
{
    int c;
    while ((c = getchar()) != '\n' && c != EOF)
    {
    }
}
```

• Function for Valid Input

Handles only A–D inputs:

```c
char getValidAnswer()
{
    char ans;

    while (1)
    {
        printf("Your answer: ");
        scanf(" %c", &ans);
        clearBuffer();

        ans = toupper(ans);

        if (ans == 'A' || ans == 'B' || ans == 'C' || ans == 'D')
        {
            return ans;
        }

        printf("Invalid input! Please enter A, B, C, or D.\n");
    }
}
```

• Score Calculation Logic

```c
if (answer == 'B')
{
    score++;
    printf("Correct!\n");
}
else
    printf("Wrong! Correct answer: B\n");
```

6.Testing & Results

Test Case 1: Valid Answer Input

Input:

```
PS C:\Users\ASUS\Desktop\c projet> cd "c:\Users\ASUS\Desktop\c projet\" ; if ($?) { gcc command_line_
quiz_game.c -o command_line_quiz_game } ; if ($?) { .\command_line_quiz_game }

========== COMMAND LINE QUIZ GAME ==========

Q1) What is the capital of India?
A) Mumbai
B) New Delhi
C) Kolkata
D) Chennai
Your answer: A
```

Output:

```
Wrong! Correct answer: B
```

Test Case 2: Invalid Input

Input:

```
Q2) Which language is used for system programming?
A) Python
B) Java
C) C
D) HTML
Your answer: 5
```

Output:

```
Invalid input! Please enter A, B, C, or D.
Your answer:
```

Test Case 3:For Right Answer

Input:

```
Q5) Who is known as the Father of Computers?
A) Charles Babbage
B) Alan Turing
C) Bill Gates
D) Tim Berners-Lee
Your answer: A
```

Output:

```
Correct!
```

7. Conclusion & Future Work

Conclusion

The Command-Line Quiz Game successfully demonstrates:

- Modular programming
- Input validation
- Character handling
- Logical decision-making
- Interactive user-based applications

The project ensures that only valid inputs are processed and provides clear feedback to the user. It is a simple yet effective demonstration of C programming concepts suitable for beginners.

Future Enhancements

The quiz game can be improved with:

- Randomized questions
- Loading questions from a file
- Timer-based quiz

- Multiple difficulty levels
- Score saving using file handling
- User login system
- High-score leaderboard

8.References

. Kernighan, B. W., & Ritchie, D. M. (1988). *The C Programming Language (2nd ed.)*. Prentice Hall.
– Classic reference for C syntax, functions, and standard libraries used in this project.

. **Herbert Schildt (2017).** *C: The Complete Reference (4th ed.)*. McGraw-Hill Education.
– Covers console input/output, ctype.h, buffer handling, and modular programming.

. **Reema Thareja (2015).** *Programming in C*. Oxford University Press.
– Explains loops, decision-making, functions, and arrays—core concepts used in quiz logic.

. **GeeksforGeeks. (n.d.).** *C Programming Language Tutorials*.
– Helpful for understanding toupper(), input validation, and character handling.

. **TutorialsPoint. (n.d.).** *C Standard Library – ctype.h*.
– Covers character classification and conversion used for validating answers.

. **J. Liberty (2020).** *C Programming and ANSI C Standards*. Pearson.
– Explains ANSI C features used in structured program design.

. **M. Banahan, D. Brady, & M. Doran (1988).** *The C Book*. Addison Wesley.
– Reference for procedural programming and console-based applications.

. **W3Schools. (n.d.).** *C Input/Output*.
– Supports understanding of scanf(), buffers, and safe input reading.

. **Sedgewick, R. (1990).** *Algorithms in C*. Addison-Wesley.
– Provides insight into flow control and algorithmic thinking used in the quiz logic.

. **Stack Overflow Community. (n.d.).** Discussions on safe input handling and buffer clearing in C.
– Practical solutions implemented in clearBuffer() and input sanitization.